Amendments to the Claims:

This listing of claims will replace all prior versions, and listings of claims in the application. Applicant has submitted a new complete claim set showing marked up claims with insertions indicated by underlining and deletions indicated by strikeouts and/or double bracketing.

Listing of Claims:

1-29. (Previously Cancelled)

30. (Previously presented) A method for processing an extensible mark up language (XML) document comprising:

parsing the XML document into a stream of schema elements and data elements;

receiving the stream of schema elements using an API;

converting the stream of schema elements into data type definition (DTD) objects;

validating the stream of data elements using the DTD objects; and

if valid, passing the stream of data elements to an application using the API.

31. (Previously presented) The method of claim 30, wherein the converting comprises:

calling a method in a first application program interface (API); and

as a result of calling the first method, calling one or more methods in a second API to construct the DTD objects.

32. (Previously presented) The method of claim 30, wherein the converting comprises referencing one or more tables that define the schema elements and associated functions for processing the schema elements.

33. (Previously presented) A computer-readable medium having computer-executable instruction, which when executed by a computer, performs the method of claim 30.

34. (Previously presented) An architecture for processing an extensible mark up language (XML) document comprising:

a parser to parse the XML document into a stream of elements including a stream of schema elements and a stream of data elements;

a converter to convert the stream of schema elements into data type definition (DTD) objects using an API and to validate the stream of data elements using the DTE objects; and

a schema node factory to pass valid data elements to an application using the API.

35. (Previously presented) The architecture of claim 34, further comprising a schema builder that utilizes one or more tables to process the elements, the tables containing information defining a schema for the XML data.

36. (Previously presented) A computer implemented with the architecture of 34.

37. (Previously presented) A client-server system, comprising:

a server;

a client connectable to the server to exchange extensible mark up language (XML) documents;

at least one of the client and the server implementing the architecture of claim 34.

38. (Previously presented) The architecture of claim 34, further comprising a validation node factory to evaluate whether the data elements comply with constraints set forth in the DTD objects.

39. (Previously presented) A system for processing an extensible mark up language (XML) document comprising:

means for parsing the XML document into a stream of schema elements and a stream of data elements;

means for converting the schema elements into data type definition (DTD) objects using an API;

means for validating the data elements using the DTD objects; and

if valid, means for passing valid data elements to an application using the API.

40. (New) The method of claim 30 wherein the converting step comprises:

obtaining a parameter from a static table; and

calling a function to construct a DTD object based on the schema element, the function constructing the DTD object based on the parameter from the static table.

41. (New) The method of claim 40 wherein the function comprises at least one of a function for signaling a new schema is being processed, signaling that an end of a schema has been reached, determining if it is valid to have text in a current position of a

schema document, determining if all content for a current schema element has been corrected provided, and creating a DTD content model node for describing a default value.

42. (New) The method of claim 40 wherein the step of calling the function includes:

identifying a new schema element in the static table;

pushing the new schema element onto a stack;

initializing at least one data structure for the new schema element; and

creating an empty DTD object for the new schema element.

43. (New) The method of claim 40 wherein the static table includes an attribute table for providing an attribute of a schema element and wherein the step of calling the function includes parsing the attribute table for an attribute value corresponding to the schema element and calling a function corresponding to the attribute value.

44. (New) The method of claim 40 wherein the static table is organized in a hierarchical arrangement and includes a schema root table at a top of the hierarchical arrangement, the schema root table including a reference to a subordinate table for describing a type of the schema element.

45. (New) The method of claim 44 wherein the subordinate table includes at least one of an element table and an attribute table, the element table including at least one field listing a type of element permitted in the schema element and the attribute table including at least one field listing a type of attribute in the schema and a corresponding function for handling the attribute.

46. (New) The method of claim 45 wherein the static table includes the element table, the element table further referencing at least one function table and wherein the at least one function table contains functions pertaining to node events.

47. (New) The method of claim 30 wherein the converting step comprises:

obtaining a parameter from a static table; and

calling a function to construct a DTD object based on the schema element including:

identifying a new schema element in the static table;

pushing the new schema element onto a stack;

initializing at least one data structure for the new schema element; and

creating an empty DTD object for the new schema element,

wherein the function constructing the DTD object is based on the parameter from the static table and including one of signaling a new schema is being processed, signaling that an end of a schema has been reached, determining if it is valid to have text in a current position of a schema document, determining if all content for a current schema element has been corrected provided, creating a DTD content model node for describing a default value,

and wherein the static table is organized in a hierarchical arrangement including a schema root table at a top of the hierarchical arrangement, an attribute table for providing an attribute of a schema element including at least one field listing a type of attribute in the schema and corresponding functions for handling the attribute, and an element table including at least one field listing a type of element permitted in the schema element, the element table further referencing at least one function table containing a function pertaining to node events,

and wherein the step of calling the function includes parsing the attribute table for an attribute value corresponding to the schema element and calling a function corresponding to the attribute value.

48. (New) The architecture of claim 34 wherein the converter obtains a parameter from a static table and calls a function to construct a DTD object based on the schema element, the function constructing the DTD object based on the parameter from the static table.

49. (New) The architecture of claim 48 wherein the converter further identifies a new schema element in the static table, pushes the new schema element onto a stack, initializes at least one data structure for the new schema element, and creates an empty DTD object for the new schema element.

50. (New) The architecture of claim 34 wherein the converter further obtains a parameter from a static table and calls a function to construct a DTD object based on the schema element including:

identifying a new schema element in the static table;

pushing the new schema element onto a stack;

initializing at least one data structure for the new schema element; and

creating an empty DTD object for the new schema element,

wherein the function to construct the DTD object is based on the parameter from the static table and including one of signaling a new schema is being processed, signaling that an end of a schema has been reached, determining if it is valid to have text in a current position of a schema document, determining if all content for a current

schema element has been corrected provided, creating a DTD content model node for describing a default value,

and wherein the static table is organized in a hierarchical arrangement including a schema root table at a top of the hierarchical arrangement, an attribute table for providing an attribute of a schema element including at least one field listing a type of attribute in the schema and corresponding functions for handling the attribute, and an element table including at least one field listing a type of element permitted in the schema element, the element table further referencing at least one function table containing a function pertaining to node events,

and wherein the step of calling the function includes parsing the attribute table for an attribute value corresponding to the schema element and calling a function corresponding to the attribute value.